ROBOFUN.RO

LECȚIA V

CURS GRATUIT

ARDUINO ȘI ROBOTICĂ

Senzori accelerație

Proiect :

Accelerometru pentru automobil

Senzori Acceleratie

Senzorii de acceleratie detecteaza (destul de evident :), acceleratia. Se pot folosi pentru masurarea acceleratiilor instantanee pe care le inregistreaza un element in miscare, sau pentru a detecta directia verticala (pe baza acceleratiei gravitationale g, care are intotdeauna directia verticala). Orice smart-phone are deja incorporat un accelerometru (pe langa multe alte dispozitive), care accelerometru este folosit pentru rotirea automata a ecranului atunci cand intorci telefonul. Atunci cand rotesti telefonul, acceleratia gravitationala a Pamantului (care intotdeauna are directia verticala) isi schimba orientarea in raport cu telefonul (pentru ca telefonul se roteste), si astfel pozitia telefonului este detectata.

De asemenea, un accelerometru este utilizat pentru a sesiza miscarea. De exemplu, in cazul unui joystick WII, accelerometrul pe 3 axe din interiorul acestuia simte miscarea mainii si misca jucatorul de pe ecran in consecinta (ca o mica paranteza, un pic mai tarziu vom vedea cum putem conecta direct la Arduino joystick-ul WII).

Din punct de vedere al conectarii la Arduino, exista doua tipuri de accelerometre : cu conectare analogica si cu conectare I2C. Cele cu conectare analogica folosesc pinii analogici ai Arduino (cate un pin pentru fiecare axa). Cele cu conectare I2C folosesc cei doi pini I2C (SDA si SCL, care in cazul Arduino sunt conectati la pinii analogici 4 si 5). Ca principiu general, accelerometrele digitale (cu conectare pe I2C) sunt mai exacte si mai putin afectate de zgomot decat cele analogice. Accelerometrele cu conectare analogica sunt insa cu mult mai simplu de folosit.

Un alt parametru al unui accelerometru este **scala** acestuia. Spre exemplu, ADXL335 poate masura de la -3g pana la 3g (are o scala de 6g). Altele, cum ar fi LIS331, au o scala selectabila din software (maxim -24g pana la 24g). Ca regula, cu cat scala este mai mica, cu atat precizia este mai mare. In functie de proiect, vei vrea sa-ti alegi un accelerometru cu o scala suficient de mare incat sa poata masura intreaga gama de acceleratii pe care le vei intalni, dar nu cu mult mai mare de atat, ca sa iti pastrezi totusi precizia.

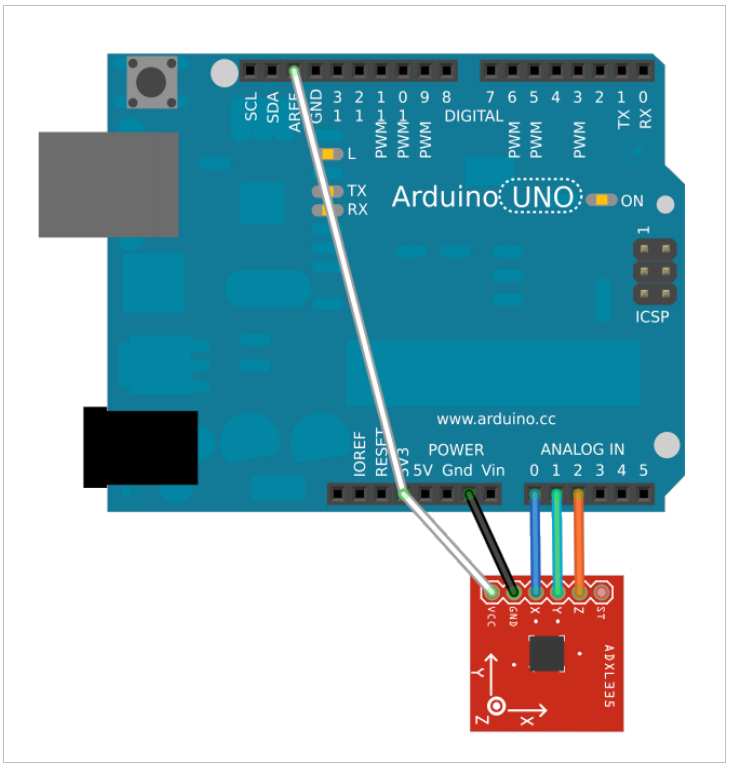
Banda unui accelerometru ("bandwidth") determina de cate ori pe secunda poate fi citit senzorul de catre Arduino. Un accelerometru cu o banda de 100 Hz poate fi citit de 100 de ori pe secunda.

Din punct de vedere al **numarului de axe** citite, exista accelerometre cu o axa, cu doua, sau cu trei axe. Cele cu trei axe sunt cel mai des intalnite.

ADXL335

ADXL335 este unul dintre cele mai simplu de utilizat accelerometre pe 3 axe. Este un accelerometru analogic, ceea ce inseamna ca informatia este transmisa catre Arduino sub forma unui semnal analogic a carui tensiune variaza direct proportional cu acceleratia. ADXL335 poate masura acceleratii in gama +/- 3 g (adica de trei ori mai mult decat acceleratia gravitationala obisnuita). Conectarea la Arduino este foarte simpla. Se conecteaza pinul VCC al ADXL335 la pinul 3.3 V Arduino (ADXL335 functioneaza la 3.3V si nu la 5V, cum am intalnit pana acum), se conecteaza pinul GND al ADXL335 la pinul GND al Arduino, si se conecteaza pinii X, Y si Z ai ADXL335 la trei pini analogici ai Arduino (sa spunem pinul analogic 0, 1 si 2). In afara acestor pini mai ramane o singura conexiune de facut – pinul 3.3 al Arduino la pinul AHREF al Arduino - . In acest mod, valorile date pe

pinii X, Y si Z de catre ADXL335 sunt raportate la tensiunea maxima de 3.3 V.



Arduino 3.3 V	ADXL335 VCC
Arduino GND	ADXL335 GND
Arduino Analog0	ADXL335 X
Arduino Analog1	ADXL335 Y
Arduino Analog2	ADXL335 Z
Arduino 3.3	Arduino AREF

```
void setup() {
 Serial.begin(9600);
 analogReference(EXTERNAL);
}

void loop() {
  float xAcc=readAcc(0);
  float yAcc=readAcc(1);
  float zAcc=readAcc(2);

  Serial.print("Acceleratie X: ");
```

```
  Serial.print(xAcc,DEC);
  Serial.print("Acceleratie Y: ");
  Serial.print(yAcc,DEC);
  Serial.print("Acceleratie Z: ");
  Serial.print(zAcc,DEC);
  Serial.println();
  delay(50);
}

float readAcc(int port) {
 int value=analogRead(port);
 int miliVolts=map(value,0,1023,0,3300)-3300/2;
 float acc=(float)miliVolts/360;
 return acc;
}
```

Interesant in codul de mai sus este "analogReference(EXTERNAL)", in rutina *setup.* Acest lucru seteaza ca referinta de tensiune pentru toate porturile analogice acea tensiune externa pusa pe pinul AHREF (noi in acest exemplu am pus acolo 3.3 V). Mai departe, valoarea citita de pe portul analogic (intre 0 si 1023) este procesata pentru a fi transformata intr-o valoare de acceleratie, exprimata ca multiplu al acceleratiei gravitationale "g".

Ca sa testezi ca functioneaza corect, misca accelerometrul in aer. In functie de orientarea acestuia in raport cu acceleratia gravitationala (care este intotdeauna verticala), vei obtine valori diferite pe cele trei axe X, Y si Z.

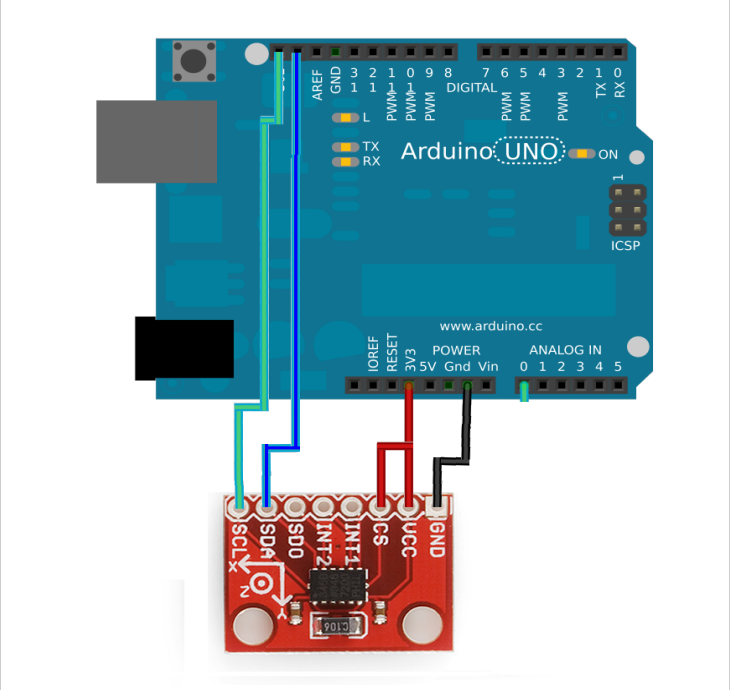
De ce nu merge ?

–un motiv des intalnit este ca ai uitat sa cuplezi firul 3.3 V la pinul AHREF (in cazul meu, cel putin, acesta este un motiv des intalnit :).

ADXL345

ADXL345 este un accelerometru digital, care se conecteaza la Arduino prin I2C. Are avantajul de a fi mult mai precis decat ADXL335 (care este analogic), si de asemenea de a fi capabil de scale dinamice care pot fi setate de catre utilizator, ceea ce ii permite o rezolutie mult mai buna.

In plus, dispune intern de un mecanism capabil sa detecteze ciocanitul (simplu sau dublu) si caderea libera. In acest mod, atunci cand accelerometrul este in cadere libera, va notifica Arduino printr-unul dintre cei doi pini INT1 sau INT2. In cele ce urmeaza ne vom ocupa doar de citirea acceleratiilor pe cele trei axe, si nu vom aprofunda regimul functionarii cu intreruperi.



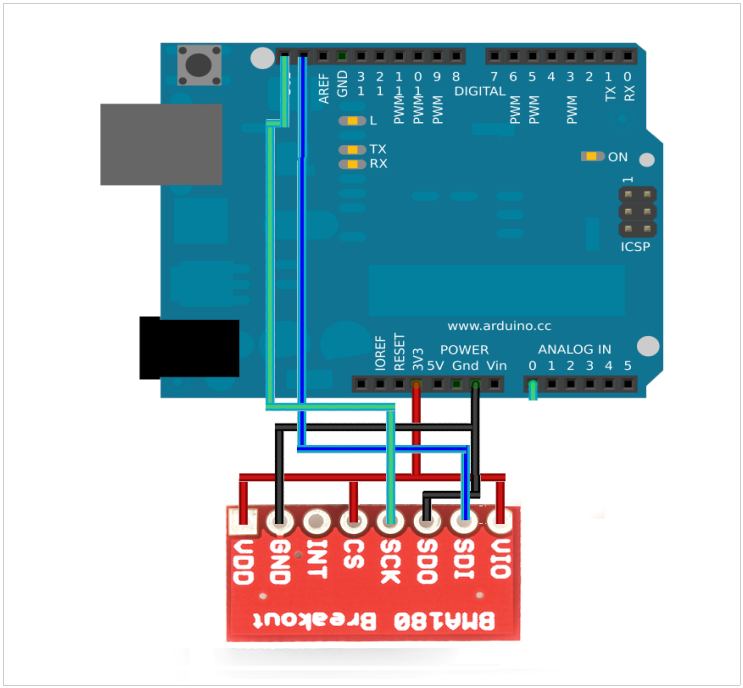
Arduino 3.3 V	ADXL345 VCC
Arduino GND	ADXL345 GND
Arduino SDA	ADXL345 SDA
Arduino SCL	ADXL345 SCL
Arduino 3.3 V	ADXL345 CS

Codul sursa este complicat, dar din fericire este deja scris integral, tot ce ai de facut este sa il utilizezi. Il gasesti pe pagina http://www.robofun.ro/accelerometru_adxl345 (urmeaza ultimul link din pagina pe bildr.org).

BMA180

BMA180 este un accelerometru foarte capabil, si extrem de configurabil. Suporta masurarea de acceleratii intr-o multitudine de game distincte – 1g, 1.5g, 2g, 3g, 4g, 8g, 16g).

Conectarea la Arduino se face folosind pinii I2C, ca in figura de mai jos.



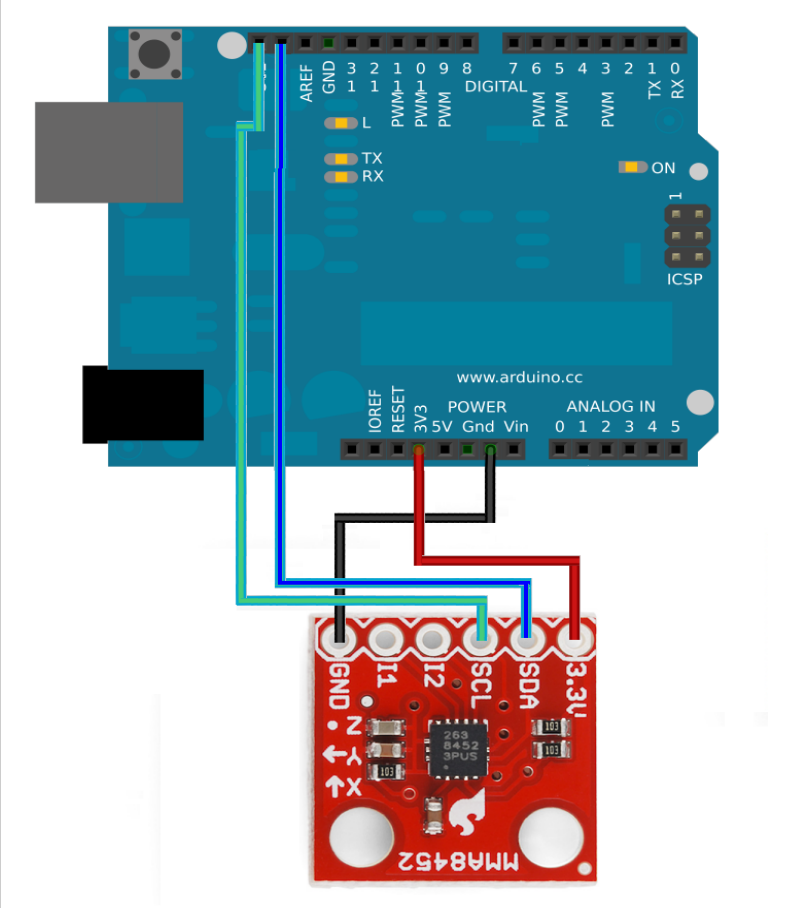
Arduino 3.3 V	BMA180 VIO
Arduino 3.3 V	BMA180 CS
Arduino 3.3 V	BMA180 VDD
Arduino GND	BMA180 SDO
Arduino GND	BMA180 GND
Arduino SDA	BMA180 SDI
Arduino SCL	BMA180 SCK

```
#include <Wire.h>
#define DEVICE ((byte)0x40)
#define DATA_X0       0x02
#define AXIS_SHIFT    2
#define DELAY_RATE    500

int counter = 0;


static void readFrom(byte address, byte num, byte *buff) {
  Wire.beginTransmission(DEVICE);
  Wire.write(address);
  Wire.endTransmission();

  Wire.requestFrom(DEVICE, num);
  num = Wire.available();

  while(num-- > 0) {
    *(buff++) = Wire.read();
  }
```

```
}

void setup()
{
  Wire.begin();
  Serial.begin(115200);
  Serial.flush();
  delay(15);
}

  int axis[5] = {0x0101, 0, 0, 0, 0};

void loop()
{

  readFrom(DATA_X0, 6, (byte*)(axis+1));

  Serial.print(millis());
  Serial.print("-");
  Serial.print(counter);
  Serial.print(":");
  Serial.print(axis[1]);
  Serial.print(", ");
  Serial.print(axis[2]);
  Serial.print(", ");
  Serial.print(axis[3]);
  Serial.println();
  counter ++;
}
```

Varianta de cod de mai sus doar citeste informatia asociata acceleratiei pe cele trei axe. Un exemplu de cod mai complex, in care este setata si gama, este disponibil aici - http://www.robofun.ro/accelerometru_bma180

MMA8452Q



Arduino 3.3 V	MMA8452Q 3.3V
Arduino GND	MMA8452Q GND
Arduino SDA	MMA8452Q SDA
Arduino SCL	MMA8452Q SCL

MMA8452Q este un accelerometru ieftin, suficient de capabil. Suporta trei game de acceleratie (2g, 4g, 8g). Conectarea la Arduino se face folosind patru fire, ca in figura de mai sus.

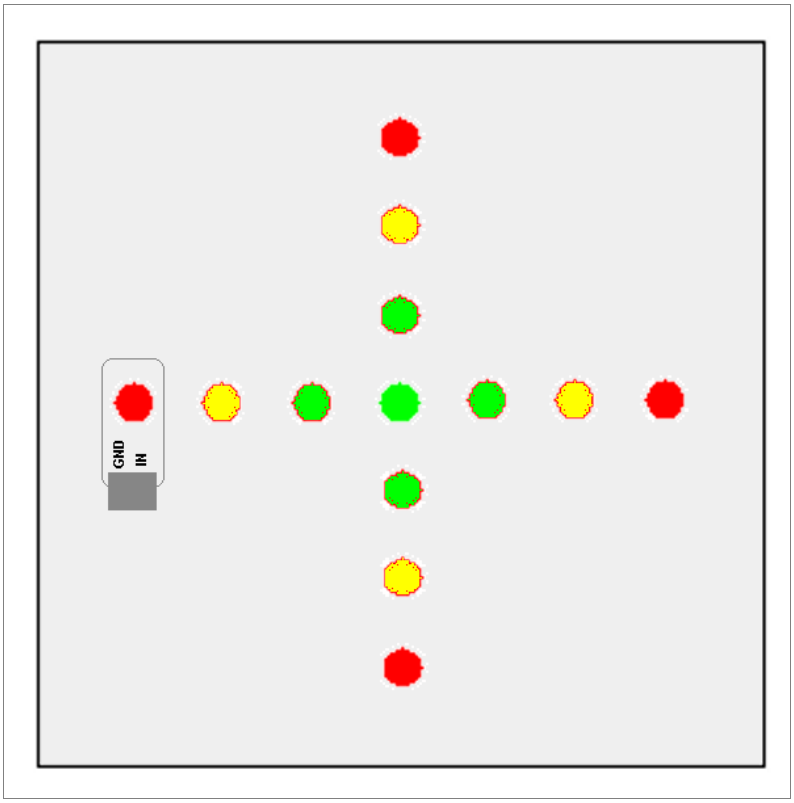
Pentru cod, exista o librarie foarte bine documentata, disponibila aici -

http://www.robofun.ro/accelerometru_MMA8452Q

Accelerometru pentru automobil

O idee relativ simpla, dar de efect, bazata pe un accelerometru, o placa Arduino si cateva led-uri (prinse sau nu in bordul masinii). Atunci cand accelerezi sau cand iei curbe cu viteza, accelerometrul detecteaza acceleratiile atat pe directia de inaintare a masinii, cat si pe axa laterala (perpendiculara pe directia de inaintare). Informatiile culese de accelerometru sunt afisate pe un panou cu led-uri. Obtii, astfel, un dispozitiv care iti ofera informatii despre acceleratia masinii tale, atat pe directia de inaintare cat si pe directie laterala.

Numarul de led-uri depinde de tine. In configuratia de mai jos, eu am ales 13 led-uri, montate in cruce. Atunci cand masina sta pe loc (sau se deplaseaza pe un drum drept, cu viteza uniforma), sta aprins doar led-ul din centru. In momentul in care accelerezi, se aprind gradual led-urile pe directia verticala, iar atunci cand iei o curba, se aprind led-urile orizontale. Ca sa fie mai spectaculos, poti monta led-uri de culoare diferita. In centru, si cele 4 led-uri care inconjoara centrul, sugerez led-uri verzi, apoi led-uri galbene, iar pe exterior led-uri rosii. Evident, la fel de simplu poti opta pentru un afisaj LCD, obtinand astfel valori numerice.



Ai mai multe variante de a monta led-urile. Spre exemplu, poti folosi led-uri brick si o placa din plexiglass negru sau gri, in care dai gauri exact pe dimensiunea led-urilor. Montezi apoi fiecare led in gaura lui (eventual pui o picatura de SuperGlue), lasand placa led-ului pe partea inferioara a placii de plexiglass. Astfel, pe partea superioara a placii de plexiglass (cea care se vede) vei avea vizibile doar 13 led-uri (pentru ca placutele led-urilor sunt prinse pe partea inferioara a placii de plexiglass si nu se vad).

Alta varianta (daca esti dispus sa faci cateva modificari in bordul masinii) ar fi sa dai cateva gauri direct in bord, si sa montezi led-urile direct, fara a mai folosi placa de plexiglas. Efectul va fi net superior, dar iti trebuie ceva curaj sa gauresti bordul masinii.

In afara de led-uri, mai ai nevoie de o placa Arduino si de un accelerometru. Poti alege orice accelerometru ai la indemana. Ideal ar fi un accelerometru cu o gama de masura in gama a cativa g, in ideea de a avea totusi precizie. Spre exemplu, ADXL335 (masoara +/- 3g), BMA180 (poate fi configurat sa masoare in gama +/- 4g), ADXL345 sau MMA8452Q (masoara in gama +/- 4g) sunt toate alegeri excelente. De departe, cel mai simplu de utilizat este ADXL335, care functioneaza in mod analogic. Din acest motiv il voi folosi mai departe in acest proiect.

Conexiunile intre componente sunt destul de simple. Singurul lucru despre care merita sa vorbim este conectarea led-urilor la Arduino. Daca ai ales sa folosesti Arduino Mega, atunci ai suficienti pini de conectare pentru led-uri, tot ce ai de facut este sa conectezi cele 14 fire de GND (13 led-uri + accelerometru) impreuna si sa le conectezi la pinul GND al placii Arduino, apoi sa conectezi pinii IN ai led-urilor la cate un pin digital al Arduino (sa spunem ca alegi pinii de la 2 la 14) si in final sa conectezi accelerometrul ADXL335 asa cum este descris in sectiunea dedicata acestuia (sa spunem ca ai ales porturile analogice 0, 1 si 2). Daca insa alegi sa folosesti un Arduino UNO, va trebui sa folosesti un mic artificiu pentru a putea conecta cele 13 led-uri la cei 12 pini digitali ai Arduino UNO (pinii de la 2 la 13, pentru ca pinii 0 si 1 sunt cei folositi pentru programarea placii si nu este bine sa ne atingem de ei). Artificiul despre care vorbesc se refera la faptul ca oricare pin analogic al placii Arduino UNO poate fi folosit si pe post de pin digital. Pur si simplu, pinul analogic 0 va fi adresat ca pinul digital 14, pinul analogic 1 va fi adresat ca pinul digital 15, si tot asa (daca vrem sa aprindem led-ul legat la pinul digital 0, vom folosi *digitalWrite(14, 0)* – dupa ce in setup am setat corect pinul – *pinMode(14, OUTPUT)).*

Recapituland, codul de mai jos corespunde urmatoarei configuratii hardware :

–Arduino UNO

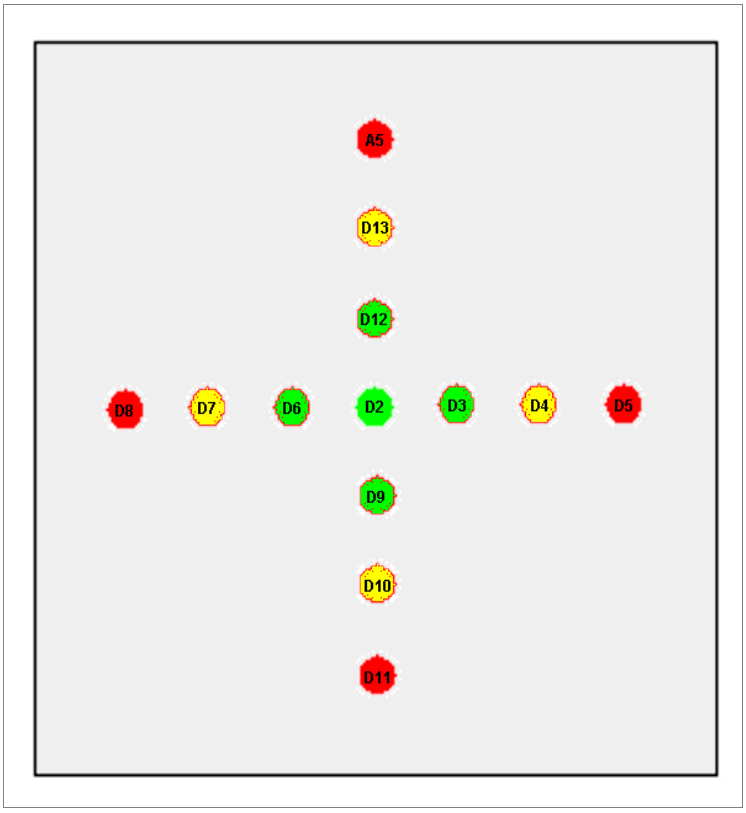
–ADXL335

–5 led-uri brick verzi

–4 led-uri brick galbene

–4 led-uri brick rosii

–conectarea led-urilor la pinii Arduino ca in poza de mai jos (led-ul din centru la pinul digital 2, led-ul din dreapta lui la pinul digital 3, led-ul de deasupra lui la pinul digital 12, si tot asa).




```
#define PRAG_X_1 0.5
#define PRAG_X_2 1
#define PRAG_X_3 2
#define PRAG_Y_1 0.5
#define PRAG_Y_2 1
#define PRAG_Y_3 2
#define SMOOTH_X 0.4
#define SMOOTH_Y 0.4

void setup(){
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(10, OUTPUT);
  pinMode(11, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(13, OUTPUT);
  //pinul analogic A5 poate fi adresat ca pin digital 19
```

```
  pinMode(19, OUTPUT);
  //led-ul din centru va sta aprins tot timpul
  digitalWrite(2, HIGH);
}

float oldAccX;
float oldAccY;
void loop() {
  float accX = smooth(readAcc(0), SMOOTH_X, oldAccX);
  float accY = smooth(readAcc(1), SMOOTH_Y, oldAccY);

  //led-urile pe directie orizontala, din dreapta
  if (accX > PRAG_X_1){
    digitalWrite(3, HIGH);
  }
  else {
    digitalWrite(3, LOW);
  }

  if (accX > PRAG_X_2){
    digitalWrite(4, HIGH);
  }
  else {
    digitalWrite(4, LOW);
  }

  if (accX > PRAG_X_3){
    digitalWrite(5, HIGH);
  }
  else {
    digitalWrite(5, LOW);
  }


  //led-urile pe directie orizontala, din stanga
  if (accX < - PRAG_X_1){
    digitalWrite(6, HIGH);
  }
  else {
    digitalWrite(6, LOW);
  }

  if (accX < -PRAG_X_2){
    digitalWrite(7, HIGH);
  }
  else {
    digitalWrite(7, LOW);
  }

  if (accX < -PRAG_X_3){
    digitalWrite(8, HIGH);
  }
  else {
    digitalWrite(8, LOW);
  }


  //led-urile pe directie verticala, din partea de sus
```

```
  if (accX > PRAG_X_1){
    digitalWrite(9, HIGH);
  }
  else {
    digitalWrite(9, LOW);
  }

  if (accX > PRAG_X_2){
    digitalWrite(10, HIGH);
  }
  else {
    digitalWrite(10, LOW);
  }

  if (accX > PRAG_X_3){
    digitalWrite(11, HIGH);
  }
  else {
    digitalWrite(11, LOW);
  }


  //led-urile pe directie verticala, din partea de jos
  if (accX < - PRAG_X_1){
    digitalWrite(12, HIGH);
  }
  else {
    digitalWrite(12, LOW);
  }

  if (accX < -PRAG_X_2){
    digitalWrite(13, HIGH);
  }
  else {
    digitalWrite(13, LOW);
  }

  if (accX < -PRAG_X_3){
    digitalWrite(19, HIGH);
  }
  else {
    digitalWrite(19, LOW);
  }

}

float readAcc(int port) {
  int value=analogRead(port);
  int miliVolts=map(value,0,1023,0,3300)-3300/2;
  float acc=(float)miliVolts/360;
  return acc;
}

float smooth(float data, float filterVal, float smoothedVal) {
  if (filterVal > 1) {
    filterVal = .99;
  }
  else if (filterVal <= 0){
```

```
    filterVal = 0;
  }
  smoothedVal = (data * (1 - filterVal)) + (smoothedVal  *  filterVal);
  return (float)smoothedVal;
}
```

Codul de mai sus citeste valoarea celor doua acceleratii (pe directia inainte si pe directia laterala) si apoi aprinde led-urilor corespunzatoare celor doua valori. Dupa citire, valorile acceleratiilor sunt filtrate trece-jos pentru a obtine un raspuns mai lent. Filtrarea se face de catre functia *smooth*. Daca eliminam functia *smooth*, vom obtine un raspuns extrem de rapid, dificil de remarcat cu ochiul liber. Printr-o filtrare trece-jos, led-urile se vor schimba mai lent. Cat de lent se schimba led-urile este controlat de valorile parametrilor SMOOTH_X si SMOOTH_Y. Cu cat valorile acestor parametrii sunt mai apropiate de unu, cu atat led-urile reactioneaza mai lent. Cu cat valorile sunt mai apropiate de zero, cu atat led-urile reactioneaza mai rapid. Nivelurile la care se aprind led-urile sunt de asemenea preconfigurate la inceputul programului (*PRAG_X_1, PRAG_X_2 ...).* Aceste prag-uri definesc nivelurile la care se aprinde si se stinge fiecare led in parte. In cazul folosirii accelerometrului ADXL335, care masoara valori ale acceleratiei intre 0 si 3 g, pragurile vor fi si ele setate intre 0 si 3.

Mai departe, tot ceea ce face codul de mai sus este sa compare valorile acceleratiilor cu pragurile prestabilite si sa aprinda sau sa stinga fiecare led. Astfel, spre exemplu, pentru led-ul conectat la portul digital 3 (imediat in dreapta led-ului din centru), avem urmatoarea secventa :

```
if (accX > PRAG_X_1){
    digitalWrite(3, HIGH);
  }
  else {
    digitalWrite(3, LOW);
  }
```

In rest, esti liber sa te joci cu valorile pragurilor si cu cele doua valori de filtrare pana cand sistemul functioneaza exact asa cum vrei tu.

O implementare interesanta a acestui gen de proiect (simplificata de faptul ca masina respectiva avea deja led-uri in bord, vei gasi aici - http://www.instructables.com/id/G-Meter/?ALLSTEPS)

Iar aici - http://www.caranddriver.com/features/the-racelogic-driftbox-feature - gasesti o implementare comerciala extinsa a principiului.

Aceasta a fost lectia 5. In final, as vrea sa te rog sa ne oferi feedback asupra acestei lectii, pentru a ne permite sa le facem mai bune pe urmatoarele.

Este vorba despre un sondaj cu 4 intrebari (oricare este optionala), pe care il poti accesa dand click aici.

Sau ne poti contacta direct prin email la contact@robofun.ro .

Iti multumim,

Echipa Robofun.RO